Multi-Level Monte Carlo for Exotic Derivative Evaluation

Anthony Maylath

February 2, 2019

1 Introduction: Pricing Exotic Financial Derivatives

Consider a financial derivative whose payoff depends on several assets. Say the product also provides an option for the holder. A popular example of such a product would be a '"Worst of Option" which has pay off:

$$P(S_T) = max\Big(\min_{s_T \epsilon S_T}(S_T) - K, 0\Big) \tag{1}$$

where S_T is a set of assets and time T represents the experation of the product. Obtaining a reasonable price for such assets is not tractable analytically. Financial Engineers typically deploy Monte Carlo simulations to evaluate such products. Typically, the approch would involve simulating a stochastic differential equation of the form:

$$ds_{i,t} = a(s_{i,t}, t)dt + b(s_{i,t}, t)dW_t \tag{2}$$

where $s_{i,t}$ is the price of asset i at time t and W_t is a Brownian Motion, possibly correlated with the other assets which determine the payoff.

2 Multi-Variable Monte Carlo

A reasonable way to solve 2 would be to discretize each $s_{i,t}$ into time steps and model dW_t with draws from a random normal. For instance, the Milstein

scheme is a popular discretation [1]. To achive quadratic convergence with the Milstein scheme, we typically need the time step to be proportional to the square of the space step, which can result in a dense grid.

A standard method to improve cost leverages the linearity of expectation, $E[p_1] = E[p_0] + E[p_1 - p_0]$, where p_1 is the price of our complex derivative and p_0 is the price of a simple derivative highly correlated to p_1. Since p_0 is simple, we do not need a dense grid to estimate it. Likewise, we can allow a sparse grid as $p_1 - p_0$ is small relative to p_0. The approch is known as a control variate.

Even with the improvement, the complexity of exotic derivatives generally require a dense grid to acheve reasonable convergence to he expectation. Such a grid can require heavy computing resources to evaluate in a fast pace trading enviornment.

Michael Giles et al. [2] have propsed an extension to the standard control variate which uses several control variates:

$$E[p_L] = E[p_0] + \sum_{l=1}^{L} E[p_l - p_{l-1}] \tag{3}$$

To evaluate an exotic derivative in the single control variable case, evaluate a portfolio of vanilla derivatives with similar behavior to the exotic product for little cost using analytical methods. The portfolio of vanillas are deployed as the control variate. We could then assign each path of the evauation to a seperate processor to solve the problem with high computing resources.

Say we want to evaluate the exotic derivative at more than one point over the input space. For the inner valuations of the inputs, we could use the neighboring points as control variates. The approch is called Multi-Level Monte Carle (MLMC). However, when we use neighboring paths in the valuation, we can no longer dispatch our valuations to run on seperate processors independantly. As a resolution, message passing software could be leveraged to comunicate control variates from neighboring points in the grid.

The method starts with evaluations on a fine grid and gradually moves to a course grid as more control variates are added. The result tends to be many evaulations on a course grid, with only a few evaluations on a fine grid. Load balancing can be tricky in such a situation as the overhead of scheduling can cost more than the benefit from parallelization for small jobs. Scheduling can become so complex that it may consume more developer time than writing the code to evaluate the expectation [3].

3 Implementation on High Performance Machines

Gmeiner, Drzisgn, et al. [3] explore the performance of MLMC with multigrid strategies on supercomputing resources. The paper applies the MLMC to uncertainty quantification for solutions to 3D partial differential equations. The problem is a bit different than the evaulation of financial derivatives, but the paper provides detailed performance results useful enough to understand how the method could perform on other applications.

The paper deploys the peta-scale German supercomputer, JUQUEEN, which features 28,672 nodes with 16 GB memory and 16 1.6 Ghz cores. As of November, 2017, JUQUEEN was ranked 22 on the supercomputer top 500 list. For a grid of $1.1 * 10^9$ mesh nodes and three levels, the authors found the time to resolve each of the levels was between two and three minutes with static scheduling. WIth the addition of each control variate the compute time went up by a few seconds.

As the number of computing resources increased, the authors found the computation time declined at a sub-linear rate. The marginal gain declined as overhead from load balancing small tasks began to outweigh the benefit of more processors. The paper deployed between 2,048 and 131,072 processors to obtain the results.

References

[1] Paul Glasserman, *Monte Carlo Methods in Financial Engineering*. Springer Science + Business Media, Inc. 233 Spring Street, New York, NY 10013 USA, 2003.

[2] Michael B. Giles, *Multilevel Monte Carlo Path Simulation*. Operations Research, Vol. 56, No. 3, May-June 2008, pp. 607-617.

[3] Bjorn Gmeiner, Dainel Drzisgn, Ulrich Rude, and Robert Scheichl, *Scheduling Massively Parallel Multigrid for Multilevel Monte Carlo Methods*. SIAM Journal on Scientific Computing, July 2016.